Hindawi
Security and Communication Networks
Volume 2018, Article ID 2492956, 10 pages
https://doi.org/10.1155/2018/2492956

Research Article

An Evolutionary Computation Based Feature Selection Method for Intrusion Detection

Yu Xue,[1,2] **Weiwei Jia**,[1] **Xuejian Zhao**,[3] **and Wei Pang**[4]

[1]*School of Computer and Software, Nanjing University of Information Science & Technology, Nanjing 210044, China*
[2]*Jiangsu Engineering Research Center of Communication and Network Technology, Nanjing University of Posts and Telecommunications, China*
[3]*School of Modern Posts, Nanjing University of Posts and Telecommunications, Nanjing 210044, China*
[4]*School of Natural and Computing Sciences, University of Aberdeen, AB24 3UE, UK*

Correspondence should be addressed to Yu Xue; xueyu@nuist.edu.cn

Received 27 June 2018; Accepted 23 September 2018; Published 9 October 2018

Guest Editor: Weizhi Meng

As the important elements of the Internet of Things system, wireless sensor network (WSN) has gradually become popular in many application fields. However, due to the openness of WSN, attackers can easily eavesdrop, intercept, and rebroadcast data packets. WSN has also faced many other security issues. Intrusion detection system (IDS) plays a pivotal part in data security protection of WSN. It can identify malicious activities that attempt to violate network security goals. Therefore, the development of effective intrusion detection technologies is very important. However, many dimensions of the datasets of IDS are irrelevant or redundant. This causes low detection speed and poor performance. Feature selection is thus introduced to reduce dimensions in IDS. At the same time, many evolutionary computing (EC) techniques were employed in feature selection. However, these techniques usually have just one Candidate Solution Generation Strategy (CSGS) and often fall into local optima when dealing with feature selection problems. The self-adaptive differential evolution (SaDE) algorithm is adopted in our paper to deal with feature selection problems for IDS. The adaptive mechanism and four effective CSGSs are used in SaDE. Through this method, an appropriate CSGS can be selected adaptively to generate new individuals during evolutionary process. Besides, we have also improved the control parameters of the SaDE. The K-Nearest Neighbour (KNN) is used for performance assessment for feature selection. KDDCUP99 dataset is employed in the experiments, and experimental results demonstrate that SaDE is more promising than the algorithms it compares.

1. Introduction

Wireless sensor networks (WSNs) are typical distributed sensor networks, which can realize data acquisition, processing, and transmission. It can monitor, perceive, and collect data from various sources or monitoring objects in the areas covered by the network and transmit them to users after data processing. As an emerging infrastructure for the application of Internet of Things, WSNs are widely used, for example, environmental monitoring, defense, urban management, medical applications, and other aspects [1, 2]. At present, most of the deployed WSNs collect scalar data like humidity and location. In practical applications of smart home, traffic monitoring, and medical monitoring, the wireless multimedia sensor network can process multimedia data, such as videos, audios, and images [3]. Therefore, WSNs are increasingly associated with people's usual economic and social activities. However, due to the weakness of wireless links, the lack of physical protection of nodes, and the dynamic nature of topology, WSNs are facing a variety of data security risks. The openness of WSNs allows attackers to easily eavesdrop, intercept, and tamper with packets. The most common attacks are denial of service attacks, Hello flooding attacks, replay routing attacks, and so on [4, 5]. These attacks may leak data and cause security problems in WSNs. Users are less likely to use large-scale WSNs that lack security protection and have privacy issues. Therefore, in order to promote the wider use and development of WSN, it is very important to address the security issues of WSN.

Intrusion detection system (IDS) plays a pivotal part in data security protection of WSNs [6], which can identify malicious activities that attempt to violate network security goals. IDSs identify malicious activities by monitoring the system in real time. Once they find abnormal situation, a warning will be issued. Dorothy first proposed an abstract model of the IDS in 1987 [7], which is a real-time IDS framework. At present, various IDSs have been deployed to detect anomalies [8]. In addition, neural networks [9], particle swarm intelligence [10], differential evolution algorithm [11], and other technologies [12] have been used in IDS to improve its performance. Among them, the metaheuristic algorithms have been well used to solve the IDS problems [13–15]. At the same time, there are many studies on IDS applied to WSNs [16, 17]. In [17], the meaning and function of external signals used in WSN are defined. In addition, it realizes distributed deployment and real-time IDS by improving DCA-RT dendritic cell algorithm. A distributed network IDS which is applicable to wireless networks is put forward in [18]. It is based on the principle of classification rule induction and swarm intelligence theory. Without the need to exchange sensitive data, this system can enable effectual model training for the IDS. Nowadays, large-scale distributed intrusion detection and intrusion detection data fusion technology are the main development directions for IDS [19]. However, IDSs need to detect huge amounts of data. Actually, most of the features in the datasets are redundant or irrelevant, which can result in an increase in training time and low detection speed. In the study of network information security, it is always prominent to find out the methods that can quickly and effectively get information of destroying security from intrusion detection data. Feature selection is significant for intrusion detection, since it can reduce the time complexity of the classifier and improve the efficiency by using optimized feature subsets.

In the current big data environment, mining the knowledge contained in big data is very important for guiding practical life and applications. Feature selection has therefore become more significance [20–25]. Feature selection is applied to intrusion detection in our paper. At present, feature selection has become a hot topic in machine learning [26]. As a way of achieving dimension reduction, it selects the best feature combination rather than the whole dataset. According to the independent relationship between feature selection and classifiers, feature selection is usually separated into two groups: the filter and the wrapper [27].

A filter method is independent of any classifiers. It only considers the relevance between features and class labels. It ranks the features through the experience of statistics, information theory, and many other disciplines. Student's t-test [28] and Fisher Discriminant Ratio [29] are typical hypothesis test means in statistical techniques. Meanwhile, features can be sorted from different perspectives such as entropy or information gain [30]. This is a methodological perspective based on information theory. Amiri *et al.* [31] has put forward an improved feature selection algorithm on account of mutual information. It can effectively identify the characteristics of the attacks by calculating the mutual information. Evaluating the quality of a subset of features can also apply the correlation [32]. If the correlation between a feature subset and classification is high, but the correlation between a feature and the others in the subset is low, then this feature subset is good. Besides, distance measurement is also used for feature selection [33]. The commonly used distance measures include Euclidean distance, standardized Euclidean distance, and martensitic distance.

In Wrappers, the subsequent learning algorithm is embedded into feature selection process and the performance of algorithms is determined by testing prediction performance of the feature subset. Besides, the impact of a single feature on the final result is taken into account. The typical means include sequence forward selection (SFS) [34] and sequence backward selection (SBS) [35]. The disadvantage of SBS is that it can only add features and cannot remove features. SFS is the opposite of SBS. Both SFS and SBS use greedy strategies, which can easily fall into local optimal values. The L to R selection algorithm (LRS) [36] has been offered to deal with such problem. There are two forms of this algorithm. On the one hand, it is a null set at the beginning. The algorithm appends L features each round first and then removes R features from it. In this way, the evaluation function value is made to the best. For another, the algorithm begins with the complete set, removing R features first round and then adding L features to make the evaluation function value optimal. The sequence floating selection is developed by LRS algorithm. Compared with LRS, the distinction of the two lies in that L and R of the sequence floating selection are not fixed but will change. It includes Sequence Floating Forward Selection (SFFS) and Sequence Floating Forward Selection (SFBS) [37]. SFFS starts from an empty set. It selects a subset x of unselected features in each round so that the evaluation function is optimal after adding subset x and then selects subset z from the selected features to optimize the evaluation function after removing subset z. SFBS is similar to SFFS, but the difference is that SFBS begins with the complete set. It removes features first in each round and then adds features.

Traditional filter and wrapper methods individually evaluate and select subsets. However, some features are not independent, but they play a great performance when they work with each other. Thus, the traditional method is not very good in this respect. Evolutionary computing (EC) methods have already been used for feature selection and classification in virtue of its overall optimization capabilities [38, 39], for instance, Particle Swarm Optimization (PSO) [40–43], Genetic Algorithm [44–46], ant colony optimization [47, 48], and some of the algorithms mentioned in [49], whereas the solution space of the feature selection problem increases exponentially with the rise of the dimension of the dataset. Therefore, more and more features lead to huge solution space. Also, a large number of uncorrelated or redundant features produce many local optima in a large solution space. Therefore, most EC methods still have local optimal stagnation problems [50]. Another reason for this problem may be that many of these methods lack the ability to explore and utilize search spaces in an appropriate manner [51]. Therefore, the applicable search methods should be automatically used based on the specific feature selection

problems. However, many existing evolutionary algorithms have only one search strategy and cannot effectively deal with the complex situations that arise in real-world problems. In other words, in many existing feature selection algorithms, only one Candidate Solution Generation Strategy (CSGS) is used to generate a new solution. In addition, IDSs need to address large-scale issues. Recently, EC methods using adaptive mechanisms have been exploited to deal with continuous optimization issues, and the performance is promising [52–56]. The adaptive mechanism is rarely used for feature selection in IDS. Therefore, a self-adaptive differential evolution (SaDE) [57] method with several CSGSs are introduced to cope with the issue of feature selection for IDSs. In SaDE, an adaptive mechanism is introduced to DE algorithm and improve its control parameter. DE is an effective method, and mechanism can increase the diversity of solutions. Combining these two can search the optimal strategy for current problem dynamically during the search process.

The remainder of this paper is organized as follows. In Section 2, the SaDE algorithm is presented. Section 3 introduces the experiment and gives results of the discussions. Conclusions and the future research work are provided in Section 4.

2. Self-Adaptive Differential Evolution

2.1. Initialization and DE Algorithm. The DE method is on account of evolutionary theory. As a heuristic random search method in view of group difference, the basic idea stems from the competitive strategy of the survival of the fittest in Darwin's theory of biological evolution. According to the differential vector between the parent's individuals, DE performs mutation, crossover, and selection operations. The algorithm contains the following aspects.

2.1.1. Initialization and Updating Mechanisms in DE. Unlike traditional initialization methods, this paper uses a mixed initialization strategy. Most particles are initialized with a few features, and the remaining particles are initialized with a large subset of features. It has been demonstrated in [50] that this initialization strategy can greatly improve the selection performance. *pbest* represents the best value of single particles. *gbest* represents particles' global best value. *Pbest* and *gbest* are updated according to their classification performance.

2.1.2. Mutation. The DE algorithm implements the mutation operation by the difference method. Random selection of two diverse individuals in a group and scaling vector differences are the common difference strategy. Afterwards, the vector is synthesized with the individual to be mutated. Formula (1) is used to generate a new individual.

$$V_i(g+1) = X_{r_1}(g) + F\left(X_{r_2}(g) - X_{r_3}(g)\right) \quad (1)$$

where g represents the g_{th} iteration of evolution. i, r_1, r_2, and r_3 are random integers of $[1, 2, \ldots, ps]$. ps represents particles' number. Moreover, $i \neq r_1 \neq r_2 \neq r_3$. F is called scaling factor, which is used to scale difference vector. It is a constant. $x_i(g)$ represents the i_{th} individual in the g_{th} generation population. $V_i(g+1)$ is the newly generated particle in the next generation. Through mutation, a new intermediate population $\{V_i(g+1),\ i = 1, 2, \ldots, ps\}$ is finally generated.

2.1.3. Crossover. Crossover aims to select individuals randomly, because DE is also a random algorithm. The crossover operations are performed between $X_i(g)$ and $V_i(g+1)$. The trial vector is generated according to formula (2).

$$U_{i,j}(g+1) = \begin{cases} V_{i,j}(g+1), & if\ rand \leq CR\ or\ j = j_{rand} \\ X_{i,j}(g), & otherwise \end{cases} \quad (2)$$

where CR is called crossover probability. It is a random value between 0 and 1. j_{rand} is a random integer of $[1, 2, \ldots, D]$. D represents the dimensions. A new individual $U_i(g+1)$ is randomly generated from a probability distribution. The reason for doing such an operation is to ensure that at least one component of $U_i(g+1)$ is contributed by the corresponding component in $V_i(g+1)$. Other variables have the same explanation as mentioned above.

2.1.4. Selection. DE adopts a greedy choice strategy. On the basis of fitness value, better individuals are selected as new ones of new population. Formula (3) below is used for selecting. Among them, f is the fitness function. Other variables have the same explanation as mentioned above.

$$X_i(g+1) = \begin{cases} U_i(g+1), & if\ f(U_i(g+1)) \leq f(X_i(g)) \\ X_i(g), & otherwise \end{cases} \quad (3)$$

2.2. Representation of Solutions. In this paper, feature selection is transformed into combinatorial optimization problems of "0" and "1", with "0" meaning not selecting the corresponding feature and "1" otherwise. The binary string is used to represent the solution. The string dimension set to D dimensions is the same as the total amount of the feature. Threshold θ is used to limit the vector range for each dimension to between 0 and 1. That is to say, if the value of the d_{th} dimension of the position is greater than θ, the corresponding value in the binary vector will be set to 1, which means choosing the d_{th} feature. Otherwise, it will be set to 0.

2.3. The Self-Adaptive Mechanism. The main goal of this mechanism is to generate the probabilities of CSGSs on account of their performance and to choose the suitable CSGS for every particle on account of these probabilities. CSGSs which have been successfully used in recent generations will be in higher probability to be selected in future generations. When a CSGS does not work well, it should be replaced by another CSGS that has good performance. We will give a brief introduction to the mechanism.

The 4 CSGSs used in our paper are assigned the initial probability. During the evolution process, the probability changes. Let p_q represents the selection probability of the q_{th} strategy, where q = 1, 2, 3, . . . , Q. Q is the number of CSGSs used, and in this research, Q =4. Then, the initial probability of each CSGS is set to be 1/4. The sum of these probabilities is 1, and p_q is recalculated according to the performance of CSGS in producing new solutions. In this research, the roulette wheel technique is applied to choose CSGSs because it can randomly select targets with high probabilities in each cycle [58]. Subsequently, the selected CSGS is applied to the corresponding particle for generating the candidate solution. The candidate solution is then evaluated and the update mechanism described in the second part is used to determine whether *pbest* and *gbest* should be updated. The $nsFlag_{i,q}$ and $nfFlag_{i,q}$ ($(i = 1, 2, \ldots, ps,\ q = 1, 2, \ldots, Q)$, where ps is the number of particles and Q has mentioned above) in the binary matrices $nsFlag_{ps\times Q}$ and $nfFlag_{ps\times Q}$ are used to record the information that reflects the relationship between the generated solution and the corresponding *pbest*. In other words, supposing that newly generated solution is preferable than the old one, afterward, $nsFlag_{i,q}$ = 1. Otherwise, $nfFlag_{i,q}$= 1. When a generation starts, $nsFlag_{ps\times Q}$ and $nfFlag_{ps\times Q}$ are initialized to $ps \times Q$-dimensional zero matrices.

In the evolution process, the i_{th} particle selects the q_{th} strategy to produce new solutions. Supposing that newly generated solution is preferable than the old one, afterwards, the corresponding position of the q_{th} strategy used by the i_{th} particle in matrix $nsFlag_{ps\times Q}$ is set to 1, which is $nsFlag_{i,q}$ = 1. Otherwise, the corresponding position in $nfFlag_{ps\times Q}$ is set to 1, that is, $nfFlag_{i,q}$ = 1. After repeated evolution for the LP generations, *nsFlag* and *nfFlag* are reinitialized to record the information in the following generation. When the evolution of the present generation is completed, all rows in *nsFlag* and *nfFlag* will be merged and the results will be recorded in $S_{k,q}$ ($k = 1, 2, \ldots, LP, q = 1, 2, \ldots, Q$, where LP is the number of generations, k is the k_{th} generation for each LP generations) and $F_{k,q}$, respectively. In other words, $S_{k,q}$ records the number of the new solutions that are produced by the q_{th} CSGS and succeed in entering into the following generation. Correspondingly, $F_{k,q}$ records the amount of the new solutions produced by the q_{th} CSGS that fail to enter into the next generation. After the evolutionary process is repeated for LP generations, all the elements of $S_{k,q}$ and $F_{k,q}$ make up the matrix $S_{LP\times Q}$ and $F_{LP\times Q}$, respectively. The strategy selection probabilities of the CSGSs are recalculated based on the statistical data stored in matrices S and F. Both $S_{LP\times Q}$ and $F_{LP\times Q}$ are initialized to be a $LP \times Q$-dimensional zero matrix at the first generation of each LP generations. After repeating the evolution of the LP generations, we can obtain the success and failure information of the CSGSs. The following steps are used to recalculate the probability of the q_{th} $(q = 1, 2, \ldots Q)$ strategy.

$$S_q^1 = \sum_{k=1}^{LP} S_{k,q} \tag{4}$$

$$S_q^2 = \begin{cases} \varepsilon, & if\ S_q^1 = 0 \\ S_q^1, & otherwise \end{cases} \tag{5}$$

$$S_q^3 = \frac{S_q^1}{\left(S_q^2 + \sum_{k=1}^{LP} F_{k,q}\right)} \tag{6}$$

$$P_q = \frac{S_q^3}{\sum_{q=1}^{Q} S_q^3} \tag{7}$$

where (4) is used to compute the sum of each column of matrix $S_{LP\times Q}$. S_q^3 is the proportion of the new solutions produced according to the q_{th} strategy and replaced their corresponding *pbests* successfully within LP generations. Meanwhile, the matrices $S_{LP\times Q}$ and $F_{LP\times Q}$ are initialized. In (5), the small value $\varepsilon = 0.0001$ is applied to avert division by 0. In other words, if $S_q^1 = 0$, then S_q^2 is equal to ε. Otherwise, S_q^2 is equal to S_q^1. The probabilities are normalized by (7) to ensure that they always sum to 1. The above steps are used to produce new probabilities for the CSGSs based on their performance during LP generations evolution. The CSGSs are chosen according to the new probabilities. Apparently, if the probability value is greater, the probability of selecting the corresponding CSGS is greater.

2.4. Candidate Solution Generation Strategy (CSGS). In our research, we use four powerful CSGSs which are inspired by mutation strategies of DE to generate new solutions [59]. They are used in the mutation operation. For simplicity, the symbol $DE/a/b$ is used to represent different mutation operators. a represents the basic vector, and b represents the number of difference vectors used. They are described as follows:

(1) The first strategy is named DE/rand/1. This has been described in formula (1).

(2) The second strategy is the generation of the next generation by the current individual, the current optimal individual, and four different random individuals. It is called DE/current-to-best/2, which is described in (8) as follows:

$$\begin{aligned} V_i(g+1) = X_i(g) &+ F\left(X_{best}(g) - X_i(g)\right) \\ &+ F\left(X_{r_1}(g) - X_{r_2}(g)\right) \\ &+ F\left(X_{r_3}(g) - X_{r_4}(g)\right) \end{aligned} \tag{8}$$

where $X_{best}(g)$ is the best individual in the g_{th} generation population. $X_i(g)$ represents the i_{th} individual in the g_{th} generation population. The meaning of other variables has been introduced previously.

(3) The third strategy is the generation of the next generation by a random individual and four different random individuals. It is called DE/rand/2, which is described as formula (9) as follows. Other variables have been mentioned before.

$$\begin{aligned} V_i(g+1) = X_{r_1}(g) &+ F\left(X_{r_2}(g) - X_{r_3}(g)\right) \\ &+ F\left(X_{r_4}(g) - X_{r_5}(g)\right) \end{aligned} \tag{9}$$

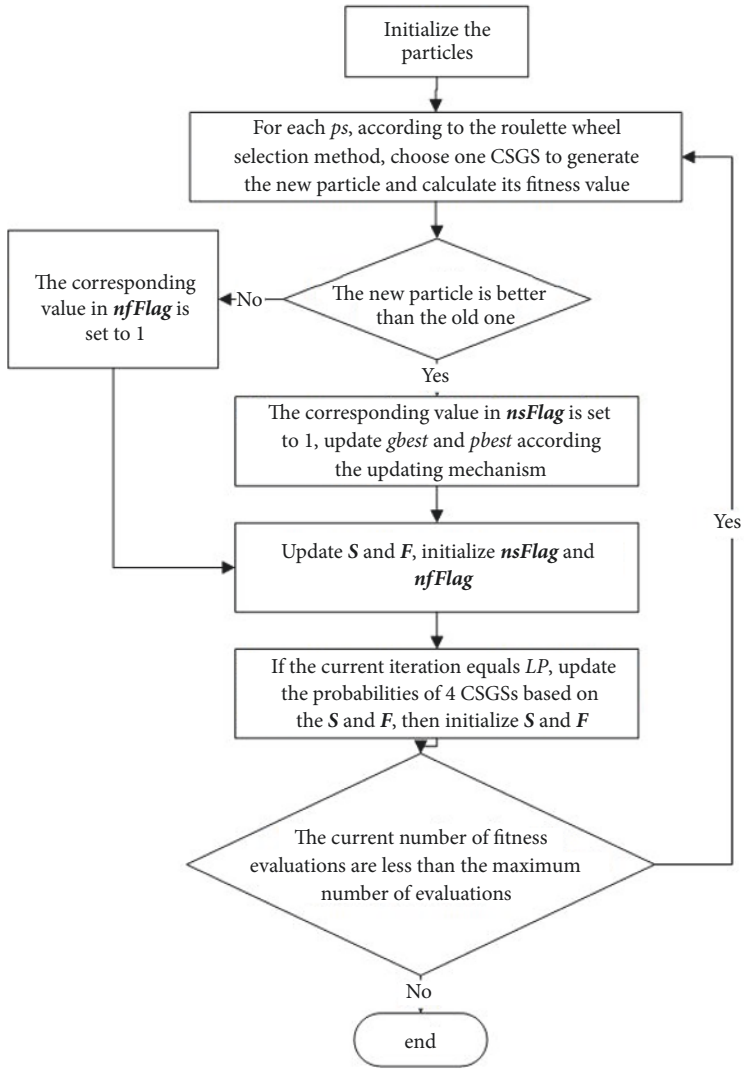

Figure 1: The flow chart of SaDE algorithm.

(4) The fourth strategy is called DE/current-to-rand/1. It includes mutation and crossover, which is described as formula (10) as follows:

$$U_i(\mathrm{g}+1) = X_i(g) + k\left(X_{r_1}(g) - X_i(g)\right) + F\left(X_{r_2}(g) - X_{r_3}(g)\right) \tag{10}$$

where k is the combination coefficient and it is a random number between 0 and 1. Other variables have been mentioned previously.

The procedure of the SaDE algorithm is shown Figure 1. The algorithm finally outputs *gbest*. The variables in the figure have been introduced in the second section.

3. Experiments and Results

The performance of the proposed method is assessed by carrying out the experiments. The sections below briefly describe the dataset, data preprocessing, parameter settings, and results of the experiments.

3.1. Datasets and Data Preprocessing. The dataset employed in this research is the KDDCUP99 dataset [60]. It is a well-known test dataset in the domain of network IDS. Each instance of this dataset has 41 feature attributes and one label. There are 13 types of content characteristics of Transmission Control Protocol (TCP) connection. There are nine types of time-based network traffic statistics and ten host-based traffic features, including four major categories and twenty-two minor categories of attacks: DoS, Probing, R2L, and U2R [61]. A number of 5 million records are included in the KDDCUP99 dataset. A 10% training subset and the test subset are offered as well. In order to save experimental time, the dataset is randomly reduced. 70% of it is used as a training set and 30% is used as a test set, in which we randomly selected 3,458 training samples and 1,482 test samples together to constitute the experimental data.

Table 1: Percentage of classification accuracies on training sets in kddcup99.

	SFFS	SBFS	Standard PSO	SaDE
Max	99.62	99.65	99.71	99.68
Min	99.28	99.42	99.68	99.68
Mean	99.48	99.58	99.68	**99.68**
Std	0.12	0.07	0.01	0.00
T-Sig	+	+	-	

Table 2: Solution sizes on training sets in kddcup99.

	SFFS	SBFS	Standard PSO	SaDE
Max	8.00	39.00	28.00	23.00
Min	2.00	37.00	14.00	11.00
Mean	**4.35**	38.42	19.81	17.69
Std	1.55	0.70	3.09	3.48
T-Sig	+	+	+	

Table 3: Percentage of classification accuracies on test sets in kddcup99.

	SFFS	SBFS	Standard PSO	SaDE
Max	98.99	98.92	98.99	98.99
Min	97.50	9818	98.45	98.38
Mean	98.60	98.64	98.68	**98.71**
Std	0.35	0.17	0.15	0.18
T-Sig	-	-	-	

Datasets are numerically processed before they are trained, as the classifier can only recognize quantitative. For the sake of testing the function of the algorithm better after improving the parameters, we also generate 4 new datasets from the KDDCUP99 dataset. Among them, we randomly selected 4 times from the original data and randomly selected 1% of the original dataset each time. We denote these datasets as DataNum1, DataNum2, DataNum3, and DataNum4, respectively. The K-Nearest Neighbour (KNN) method is applied as a classification method to evaluate subsets of features generated. In KNN, 3-fold cross validation is employed to measure the classification accuracy.

3.2. Parameter Settings. We choose SFFS, SBFS, standard PSO, and SaDE for comparison. According to past experience, each algorithm runs 26 times on the KDDCUP99 dataset. With regard to 4 CSGSs used in our paper, initial CR=0.5, F is selected from normal distribution with μ=0.5 and σ=0.3. Furthermore, ps =100. The generations of evolution named LP were empirically set to 10.

3.3. Results and Analysis. The results according to solution size and classification accuracy on the training set and the test set will be shown in the part. The solution size is the number of features chosen by the feature selection that are most beneficial to ameliorate the classification accuracy. The best result will be bold. We compare the performance of SaDE and other algorithms on DataNum1 and compare the performance of SaDE after improving the control parameter on DataNum1 to DataNum4.

Table 1 shows the classification accuracy of SaDE and other algorithms on training sets. As indicated in Table 1, the results of solution sizes are obtained by the algorithms mentioned above, including Max, Min, mean values (Mean), and standard deviations (Std). Min represents the minimum value of classification accuracy. Max means the opposite meaning of it. Mean expresses the average of the classification accuracy over 26 runs and Std shows the standard deviation in the same situation. The t-test is a statistical test used to check hypothesis with the average value of the given trust level. In our experiments DF (degree freedom) =50, and the t is equal to 2,009 (when the trust level is equal to 0.95). Therefore the results obtained are statistically important when t is less than -2,009 or higher than +2,009. We only check two cases: IMPORTANT (+) or NOT IMPORTANT (-). 'T-Sig' means the algorithm introduced in this paper is significantly distinct from other algorithms. Table 2 provides the solution sizes of the mentioned methods on training sets. Table 3 presents the classification accuracies on the test sets. According to the comparison between the SaDE and other methods, we can see that SaDE has the highest classification accuracy on test sets and training sets. Simultaneously, it has the second fewest discriminative features. Although the SFFS method has the fewest discriminative features, its amount of the selected feature is too few and its classification accuracy is poor. Moreover, the standard deviation of the classification results

Table 4: Classification accuracies of thresholds of SaDE in test sets in 4 datasets.

Dataset	0.6		0.7		0.8		0.5	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
DataNum1	98.71	0.18	98.74	0.19	98.60	0.21	**99.23**	0.13
T-Sig	+		+		+			
DataNum2	98.97	0.12	99.02	0.16	98.95	0.25	**99.05**	0.10
T-Sig	+		-		-			
DataNum3	99.09	0.19	99.15	0.22	99.17	0.14	**99.17**	0.14
T-Sig	-		-		-			
DataNum4	98.46	0.14	98.44	0.13	**98.57**	0.22	98.46	0.15
T-Sig	-		-		+			

Table 5: Classification accuracies of thresholds of SaDE in training sets in 4 datasets.

Dataset	0.6		0.7		0.8		0.5	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
DataNum1	99.68	0.18	99.68	0.19	99.66	0.21	**99.68**	0.13
T-Sig	-		-		-			
DataNum2	99.70	0.01	99.71	0.02	99.67	0.01	**99.71**	0.01
T-Sig	+		-		+			
DataNum3	99.68	0.01	99.68	0.01	99.66	0.02	**99.68**	0.01
T-Sig	-		-		+			
DataNum4	99.82	0.01	99.82	0.01	99.81	0.02	**99.82**	0.01
T-Sig	-		-		+			

of SaDE is good no matter it is in the test sets or the training sets. This result indicates that SaDE has good robustness. By comparison, although SFFS has the least characteristics, its robustness is not as good as that of SaDE. That is because adaptive mechanism and 4 CSGSs can increase the diversity of solutions. It can search the optimal strategy for current problem dynamically during the search process. Considering the statistically significant difference on the training sets, from the perspective of classification accuracy, the results obtained are statistically important between SaDE and SFFS and SBFS, and not important between SaDE and standard PSO. From the perspective of solution sizes, the results obtained are statistically important between SaDE and the other three methods. According to the difference on the test sets, the results obtained are not important between SaDE and other methods.

In addition, from these Tables 1–3, we can see that other algorithms are inferior to SaDE according to classification detection rate and the number of feature reduction. In summary, we can conclude that SaDE is an effective technique in IDS. It can also select the most useful and representative subset of intrusion detection features to reduce computational cost for IDS. From this, we can see that the adaptive mechanism and multiple CSGSs can improve the performance of the DE algorithm on IDS.

We improve the performance of SaDE by optimizing its parameters. We tested the different values of SaDE parameters in the above four datasets to test their effectiveness on the detection rate. Tables 4 and 5 show the effect of different thresholds on the classification accuracy of test sets and training sets in DataNum1 to DataNum4, respectively. The unit is the percentage. The threshold is a significant part in the initialization phase. It determines whether the features are selected. In experiments comparing SaDE with other algorithms, we set the threshold to 0.6 based on experimental experience. To improve the algorithm's performance, we set 4 different values of the threshold within the range $[0, 1]$. The reasons for this setting are briefly explained in the Section 2. From the table, we can see that, whether in the test sets or the training sets, when the threshold is set to 0.5, the classification accuracy is the best. Besides, in most cases, the robustness is also the best. Considering the statistically significant difference on the test sets and training sets, when the threshold is set to different values, the results obtained are statistically important between 0.5 and 0.8(0.6), but not significant between 0.5 and 0.7. Therefore, by optimizing the parameters, it helps improve the classification accuracy.

4. Conclusions and Future Work

Nowadays, information technology is entering the era of Internet of Things (IoTs) from the Internet age. With the application of IoTs, WSN is facing more and more data security problems. Security is a key issue in WSN design, because it seriously affects the application prospect of WSN. Intrusion detection is an important way to ensure network security. The improvement of its technology is also an aspect of guaranteeing the data security of WSN. The feature selection problem has been analyzed and the SaDE algorithm has been introduced to solve this kind of problem of IDS.

The KDDCUP99 intrusion dataset was applied to assess the performance of the introduced algorithm. Our scheme applies an adaptive mechanism in the DE algorithm to find the CSGS that is most suitable for generating new solutions. At the same time, we have improved the control parameters of SaDE. According to the results of experiments, it can be seen that the improved SaDE can effectively solve the IDS problem. On the one hand, by comparing the SaDE algorithm with other methods, we can see that the SaDE algorithm can reduce about 57% of the features in the problem. In addition, the SaDE method is superior to other algorithms in terms of classification accuracy of training sets and test sets. For another, four datasets generated from KDDCUP99 were used to test the control parameters. When the threshold is set to 0.5, the classification accuracy of SaDE is better than other values, and the performance of SaDE has been improved.

In the problems of intrusion detection, multiobjective feature selection is also a field which has been researched for many years, and SaDE algorithm has not been used in this field. Therefore, we can also resolve the multiobjective feature selection problem in intrusion detection by combining the classifier and SaDE algorithm in the future. Moreover, we can also make improvements in the initialization section.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Disclosure

This work is based on the conference paper that was presented in "The 4th International Conference on Cloud Computing and Security (ICCCS2018)".

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (61403206, 61771258, and 61876089), the Natural Science Foundation of Jiangsu Province (BK20141005 and BK20160910), the Natural Science Foundation of the Jiangsu Higher Education Institutions of China (14KJB520025), the Priority Academic Program Development of Jiangsu Higher Education Institutions, the Open Research Fund of Jiangsu Engineering Research Center of Communication and Network Technology, NJUPT (JSGCZX17001), and the Natural Science Foundation of Jiangsu Province of China under Grant BK20140883.

References

[1] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer Networks*, vol. 52, no. 12, pp. 2292–2330, 2008.

[2] W. Sun, Z. Cai, Y. Li, F. Liu, S. Fang, and G. Wang, "Security and privacy in the medical internet of things: a review," *Security and Communication Networks*, vol. 2018, Article ID 5978636, 9 pages, 2018.

[3] Z. Pan, J. Lei, Y. Zhang, and F. L. Wang, "Adaptive fractional-Pixel motion estimation skipped algorithm for efficient HEVC motion estimation," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 14, no. 1, pp. 1–19, 2018.

[4] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: Attacks and countermeasures," *Ad Hoc Networks*, vol. 1, no. 2-3, pp. 293–315, 2003.

[5] P. Li, X. Yu, H. Xu, J. Qian, L. Dong, and H. Nie, "Research on secure localization model based on trust valuation in wireless sensor networks," *Security and Communication Networks*, vol. 2017, Article ID 6102780, 12 pages, 2017.

[6] J. Granjal and A. Pedroso, "An intrusion detection and prevention framework for internet-integrated CoAP WSN," *Security and Communication Networks*, vol. 2018, Article ID 1753897, 14 pages, 2018.

[7] D. E. Denning, "An intrusion-detection model," *IEEE Transactions on Software Engineering*, vol. SE-13, no. 2, pp. 222–232, 1987.

[8] C. Khammassi and S. Krichen, "A GA-LR wrapper approach for feature selection in network intrusion detection," *Computers & Security*, vol. 70, pp. 255–277, 2017.

[9] H. I. Ahmed, N. A. Elfeshawy, S. F. Elzoghdy, H. S. Elsayed, and O. S. Faragallah, "A neural network-based learning algorithm for intrusion detection systems," *Wireless Personal Communications*, vol. 97, no. 2, pp. 3097–3112, 2017.

[10] I. Ahmad and F. E Amin, "Towards feature subset selection in intrusion detection," in *Proceedings of the 7th IEEE Joint International Information Technology and Artificial Intelligence Conference (ITAIC '14)*, pp. 68–73, December 2014.

[11] A. A. Aburomman and M. B. Ibne Reaz, "A novel weighted support vector machines multiclass classifier based on differential evolution for intrusion detection systems," *Information Sciences*, vol. 414, pp. 225–246, 2017.

[12] S. Rastegari, P. Hingston, and C.-P. Lam, "Evolving statistical rulesets for network intrusion detection," *Applied Soft Computing*, vol. 33, pp. 348–359, 2015.

[13] S. X. Wu and W. Banzhaf, "The use of computational intelligence in intrusion detection systems: a review," *Applied Soft Computing*, vol. 10, no. 1, pp. 1–35, 2010.

[14] S. Revathi and A. Malathi, "Optimization of KDD Cup 99 dataset for intrusion detection using hybrid swarm intelligence with random forest classifier," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 3, no. 7, pp. 1382–1387, 2013.

[15] Y. Y. Chung and N. Wahid, "A hybrid network intrusion detection system using simplified swarm optimization (SSO)," *Applied Soft Computing*, vol. 12, no. 9, pp. 3014–3022, 2012.

[16] I. Onat and A. Miri, "An intrusion detection system for wireless sensor networks," in *Proceedings of the IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob '05)*, pp. 253–259, Québec, Canada, August 2005.

[17] X. Xiao and R. Zhang, "Study of immune-based intrusion detection technology in wireless sensor networks," *Arabian Journal for Science and Engineering*, vol. 42, no. 8, pp. 3159–3174, 2017.

[18] C. Kolias, V. Kolias, and G. Kambourakis, "TermID: a distributed swarm intelligence-based approach for wireless intrusion detection," *International Journal of Information Security*, vol. 16, no. 4, pp. 401–416, 2017.

[19] G. Li, Z. Yan, Y. Fu, and H. Chen, "Data fusion for network intrusion detection: a review," *Security and Communication Networks*, vol. 2018, Article ID 8210614, 16 pages, 2018.

[20] R. Gurusamy and V. Subramaniam, "A machine learning approach for MRI brain tumor classification," *Computers, Materials and Continua*, vol. 53, no. 2, pp. 91–109, 2017.

[21] C. Wu, E. Zapevalova, Y. Chen et al., "Time optimization of multiple knowledge transfers in the big data environment," *Computers, Materials and Continua*, vol. 54, no. 3, pp. 269–285, 2018.

[22] S. A. Yıldızel and A. U. Öztürk, "A study on the estimation of prefabricated glass fiber reinforced concrete panel strength values with an artificial neural network model," *Computers, Materials and Continua*, vol. 552, pp. 42–51, 2016.

[23] Y. Zheng, B. Jeon, L. Sun, J. Zhang, and H. Zhang, "Student's t-hidden markov model for unsupervised learning using localized feature selection," *IEEE Transactions on Circuits and Systems for Video Technology*, 2017.

[24] Y. Zheng, L. Sun, S. Wang, J. Zhang, and J. Ning, "Spatially regularized structural support vector machine for robust visual tracking," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–11, 2018.

[25] Y. Zhang, X.-F. Song, and D.-W. Gong, "A return-cost-based binary firefly algorithm for feature selection," *Information Sciences*, vol. 418-419, pp. 561–574, 2017.

[26] Z. Yong, G. Dun-wei, and Z. Wan-qiu, "Feature selection of unreliable data using an improved multi-objective PSO algorithm," *Neurocomputing*, vol. 171, pp. 1281–1290, 2016.

[27] M. Dash and H. Liu, "Feature selection for classification," *Intelligent Data Analysis*, vol. 1, no. 1–4, pp. 131–156, 1997.

[28] W. C. Navidi, *Statistics for Engineers and Scientists*, vol. 2, McGraw-Hill, New York, NY, USA, 2006.

[29] S. Theodoridis and K. Koutroumbas, "Pattern recognition and neural networks," in *Advanced Course on Artificial Intelligence*, vol. 2049 of *Lecture Notes in Computer Science*, pp. 169–195, Springer Berlin Heidelberg, 2001.

[30] J. R. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, no. 1, pp. 81–106, 1986.

[31] F. Amiri, M. M. R. Yousefi, and C. Lucas, "Mutual information-based feature selection for intrusion detection systems," *Journal of Network & Computer Applications*, vol. 34, no. 4, pp. 1184–1199, 2011.

[32] M. M. Kabir, M. M. Islam, and K. Murase, "A new wrapper feature selection approach using neural network," *Neurocomputing*, vol. 73, no. 16–18, pp. 3273–3283, 2010.

[33] M. Liu and D. Zhang, "Feature selection with effective distance," *Neurocomputing*, vol. 215, pp. 100–109, 2016.

[34] P. Pudil and J. Hovovicova, "Novel methods for subset selection with respect to problem knowledge," *IEEE Intelligent Systems*, vol. 13, no. 2, pp. 66–74, 1998.

[35] T. Marill and D. M. Green, "On the effectiveness of receptors in recognition systems," *IEEE Transactions on Information Theory*, vol. 9, no. 1, pp. 11–17, 1963.

[36] S. D. Strearns, "On selecting features for pattern classifiers," in *Proceedings of the 3rd International Conference on Pattern Recognition (ICPR)*, 1976.

[37] P. Pudil, J. Novovičová, and J. Kittler, "Floating search methods in feature selection," *Pattern Recognition Letters*, vol. 15, no. 11, pp. 1119–1125, 1994.

[38] Y. Xue, T. Ma, B. Zhao, and A. X. Liu, "An evolutionary classification method based on fireworks algorithm," *International Journal of Bio-Inspired Computation*, vol. 11, no. 3, pp. 149–158, 2018.

[39] Y. Zhang, D.-W. Gong, J.-Y. Sun, and B.-Y. Qu, "A decomposition-based archiving approach for multi-objective evolutionary optimization," *Information Sciences*, vol. 430-431, pp. 397–413, 2018.

[40] J. Tian and H. Gu, "Anomaly detection combining one-class SVMs and particle swarm optimization algorithms," *Nonlinear Dynamics*, vol. 61, no. 1-2, pp. 303–310, 2010.

[41] B. Xue, M. Zhang, and W. N. Browne, "Multi-objective particle swarm optimisation (PSO) for feature selection," in *Proceedings of the 14th International Conference on Genetic and Evolutionary Computation (GECCO '12)*, pp. 81–88, July 2012.

[42] Y. Zhang, D.-W. Gong, and J. Cheng, "Multi-objective particle swarm optimization approach for cost-based feature selection in classification," *IEEE Transactions on Computational Biology and Bioinformatics*, vol. 14, no. 1, pp. 64–75, 2017.

[43] Y. Zhang, D. Gong, Y. Hu, and W. Zhang, "Feature selection algorithm based on bare bones particle swarm optimization," *Neurocomputing*, vol. 148, pp. 150–157, 2015.

[44] B. M. Aslahi-Shahri, R. Rahmani, M. Chizari et al., "A hybrid method consisting of GA and SVM for intrusion detection system," *Neural Computing and Applications*, vol. 27, no. 6, pp. 1669–1676, 2016.

[45] R. Li, J. Lu, Y. Zhang, and T. Zhao, "Dynamic Adaboost learning with feature selection based on parallel genetic algorithm for image annotation," *Knowledge-Based Systems*, vol. 23, no. 3, pp. 195–201, 2010.

[46] F. Souza, T. Matias, and R. Araújo, "Co-evolutionary genetic Multilayer Perceptron for feature selection and model design," in *Proceedings of the IEEE 16th Conference on Emerging Technologies and Factory Automation (ETFA '11)*, September 2011.

[47] Z. Yan and C. Yuan, "Ant colony optimization for feature selection in face recognition," in *Applied Soft Computing, Biometric Authentication*, vol. 3072 of *Lecture Notes in Computer Science*, pp. 221–226, Springer Berlin Heidelberg, 2004.

[48] N. M. O'Boyle, D. S. Palmer, F. Nigsch, and J. B. Mitchell, "Simultaneous feature selection and parameter optimisation using an artificial ant colony: Case study of melting point prediction," *Chemistry Central Journal*, vol. 2, no. 1, 2008.

[49] B. Xue, M. Zhang, W. N. Browne, and X. Yao, "A survey on evolutionary computation approaches to feature selection," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 4, pp. 606–626, 2016.

[50] B. Xue, M. Zhang, and W. N. Browne, "Particle swarm optimisation for feature selection in classification: novel initialisation and updating mechanisms," *Applied Soft Computing*, vol. 18, pp. 261–276, 2014.

[51] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artificial Intelligence*, vol. 97, no. 1-2, pp. 273–324, 1997.

[52] Y. Xue, J. Jiang, B. Zhao, and T. Ma, "A self-adaptive artificial bee colony algorithm based on global best for global optimization," *Soft Computing*, pp. 1–18, 2017.

[53] Y. Xue, Y. Zhuang, T. Ni, S. Ni, and X. Wen, "Self-adaptive learning based discrete differential evolution algorithm for

solving CJWTA problem," *Journal of Systems Engineering and Electronics*, vol. 25, no. 1, pp. 59–68, 2014.

[54] Y. Xue, B. Zhao, T. Ma, and W. Pang, "A self-adaptive fireworks algorithm for classification problems," *IEEE Access*, vol. 6, pp. 44406–44416, 2018.

[55] Y. Wang, B. Li, T. Weise, J. Wang, B. Yuan, and Q. Tian, "Self-adaptive learning based particle swarm optimization," *Information Sciences*, vol. 181, no. 20, pp. 4515–4538, 2011.

[56] C. Li, S. Yang, and T. T. Nguyen, "A self-learning particle swarm optimizer for global optimization problems," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 42, no. 3, pp. 627–646, 2012.

[57] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 398–417, 2009.

[58] D. B. Fogel, "Introduction to simulated evolutionary optimization," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 5, no. 1, pp. 3–14, 1994.

[59] J. Jiang, Y. Xue, T. Ma, and Z. Chen, "Improved artificial bee colony algorithm with differential evolution for the numerical optimisation problems," *International Journal of Computational Sciences and Engineering*, vol. 16, no. 1, pp. 73–84, 2018.

[60] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *Proceedings of the 2nd IEEE Symposium on Computational Intelligence for Security and Defence Applications (CISDA '09)*, pp. 1–6, IEEE, July 2009.

[61] Y. Zhu, J. Liang, J. Chen, and Z. Ming, "An improved NSGA-III algorithm for feature selection used in intrusion detection," *Knowledge-Based Systems*, vol. 116, pp. 74–85, 2017.